MAT 271: Applied & Computational Harmonic Analysis
Homework 2: due Monday, 05/15/23

Problem 0: Install the Julia programming language on your computer, and familiarize yourself with the Julia programming environment. I strongly suggest that you view this highly useful webpage to install the Julia system on your computer. Then, try to watch some Julia tutorials available on `YouTube.com`.

Problem 1: Let $\boldsymbol{w}_N^k := \frac{1}{\sqrt{N}}\left[\omega_N^0, \omega_N^k, \omega_N^{2k} \dots, \omega_N^{k(N-1)}\right]^{\mathsf{T}} \in \mathbb{C}^N$, where $\omega_N = \exp(2\pi\mathrm{i}/N)$. Prove

$$\left\langle \boldsymbol{w}_N^k, \boldsymbol{w}_N^\ell \right\rangle = \delta_{k,\ell},$$

where $\delta_{k,\ell}$ is Kronecker's delta and $0 \le k, \ell \le N-1$.

Problem 2: Consider a periodized versions of the function over $[-1/2, 1/2)$:

$$f(x) = ax, \quad -\frac{1}{2} \le x < \frac{1}{2}, \quad a > 0.$$

(a) Compute the Fourier coefficients c_k of this periodic function by hand.

(b) Using Julia, do the following:

1) Determine the value of a so that after the discretization of this function on a uniform grid of length 1024, the resulting vector has a unit ℓ^2-norm;

2) Apply Julia's `fft` to the input vector prepared in 1); then divide the results by `N=1024`. Note that you need to add the FFTW.jl package in your Julia session via `using Pkg; Pkg.add("FFTW")`

3) Display both the real and imaginary parts of the output vector computed in 2). I would suggest the use of the Plots.jl package for various plots;

4) Plot the hand-computed Fourier coefficients in (a) with a computed in 1);

5) Do these two ways of computing Fourier coefficients agree? What is your reasoning if they do not. Then, manipulate the input signal so that the result of the Julia `fft` followed by division by `N` *best* matches with the hand-computed Fourier coefficients used in Part 4. Throughout this problem, be as quantitative as possible.

Problem 3: Consider a periodized version (with period 1) of the following function:

$$f(x) = ax^2, \quad -\frac{1}{2} \le x < \frac{1}{2}, \quad a > 0.$$

Repeat (a), (b) of Problem 2 for this function. In addition,

(c) Compare the speed of the decay of the Fourier coefficients of this function with that of Problem 2. Which decays faster? Why?

Problem 4: Consider a periodized version (with period 1) of the following function:

$$f(x) = a\mathrm{e}^{-x^2/2\sigma^2}, \quad -\frac{1}{2} \le x < \frac{1}{2}, \quad a, \sigma > 0.$$

Repeat (1)–(3) of (b) of Problem 3 with $\sigma = 1, 0.1, 0.01$. In addition,

4) Compare the speed of the decay of the Fourier coefficients of this function with these different values of σ;

5) Compare these decays with those of Problems 2 and 3. Which decays faster? Why?

6) Do they agree with the discretized version of the Fourier transform formula of Problem 3 of HW #1 (with appropriate multiplicative constants)? If not, state your interpretation/reasoning.

Problem 5: Let us use the definition of DFT as in my lecture. Hence, given an input vector $\boldsymbol{f}$ of length N, the matrix-vector representation of the DFT applied to $\boldsymbol{f}$ is $\boldsymbol{F} = \widetilde{W}_N^* \boldsymbol{f}$.

(a) Let W_N be the DFT matrix defined in my lecture. Let D_N be the matrix representation of the Julia function `fft` so that the result of `fft` applied to the vector $\boldsymbol{f}$ of length N in Julia is $D_N \boldsymbol{f}$. Express D_N using W_N.

(b) Let S_N be the matrix representation of the Julia function `fftshift` as in my lecture. Then the Julia expression `fftshift(fft(f))` corresponds to the matrix-vector expression $S_N D_N \boldsymbol{f}$. show that $\widetilde{W}_N^* \boldsymbol{f} \neq S_N D_N \boldsymbol{f}$, and express $\widetilde{W}_N^*$ using S_N, D_N as well as the circulant-shift matrix T_N defined in my lecture.

(c) Using Julia, do the following exercise and submit the figures.

```
# Assuming you already added FFTW and Plots packages,
# let's use them!
using FFTW, Plots

# Set up the x variable [-pi, pi].
N = 16;
x = ((-N/2):(N/2-1))*2*pi/N;

# Generate a simple example function f=cos(x).
f = cos.(x);

# Do the fftshift(fft) using proper normalization.
F = fftshift(fft(f)/sqrt(N));

# Plot the real and imaginary parts of F.
plot(real(F), line=:stem, marker=:circ, label="Re(F)")
plot!(imag(F), line=:stem, color=:red, marker=:star, label="Im(F)")
```

Print this figure and submit it. You may feel the result is counterintuitive!

(d) Using the result of (b), compute $\boldsymbol{F} = \widetilde{W}_N^* \boldsymbol{f}$ where $\boldsymbol{f}$ is the same cos function as in (c). Note that you need to either use the `fftshift` function or generate the matrix S_N and do matrix-vector multiplication to obtain $\boldsymbol{F}$. Then display the real and imaginary parts using the `plot` function as before. What do you see here? You should see more intuitive results now. Submit this figure too.

Problem 6: Read the paper of the Discrete Cosine Transform by Gilbert Strang, which you can download from https://doi.org/10.1137/S0036144598336745.