EE 102A - Assignment 9

Nicholas Dwork

Problem 1. *A Conversation Between Students*

Student A: Suppose we are given a function f with compact support of length L (e.g. it is only non-zero on the interval $[-L/2, L/2]$). In the previous homework, we showed that the Fourier Transform of this function is an impulse train where the weights are equal to the Fourier Series values. Therefore, the Fourier Transform of f is 0 almost everywhere.

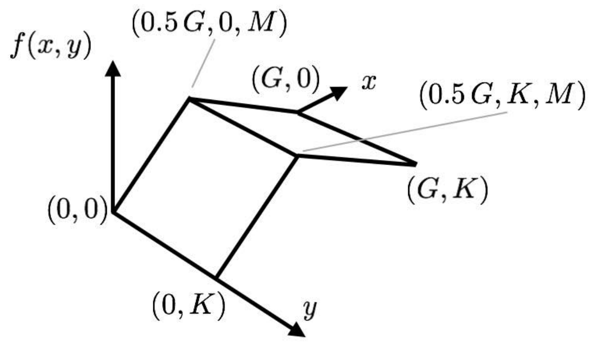
Student B: I disagree. We can model f as a new function $g : \mathbb{R} \to \mathbb{C}$ where $g(x) = \Pi(x/L)\, f(x)$. We know that if we take the Fourier Transform of g then we get $G(k) = L\,\text{sinc}(Lk) * F(k)$. Since sinc is non-zero almost everywhere, in general, we would get non-zero values almost everywhere in the Fourier domain.

Which student is correct? Explain why the other student is incorrect.

Problem 2. *Pre-filtering*
An antenna is connected to an analog-to-digital converter. The Sampling frequency of the converter is 10 kHz.
a) Design a pre-filter (out of resistors, capacitors, inductors, and operational amplifiers) to place in between the antenna and the A/D converter to avoid (i.e. significantly reduce) aliasing.
b) Plot the modulation transfer function of your filter.

Problem 3. *Multi-dimensional Fourier Transform Theorems*
Prove the following theorems:
a) Prove a Parseval's Theorem for the two-dimensional Fourier Transform. That is, show that $\|F\|_2 = \|f\|_2$, where $F = \mathcal{F}_{2D}\{f\}$.
b) A two-dimensional shift theorem (where each dimension is shifted by a different value).
c) A two-dimensional scaling theorem (where each dimension is scaled by a different value).

Problem 4. Let f be defined as shown in the figure below.



Compute the 2D Fourier transform of f for the following combinations of M, G, and K.

M	1	2	1	1	2
G	1	1	0.5	1	0.5
K	1	1	1	0.5	0.5

Problem 5. *Image Deblurring*
Recall that if x and y represent the input and output of a Linear Shift Invariant (LSI) system, then $Y = HX$, where Y and X are the Fourier transforms of y and x, and H is the transfer function of the system. If one wants to recover x given y, one might think to compute $X = Y/H$ and then inverse Fourier transform. The problem with this approach is that it is unstable for small values of H (and undefined where H is 0).

The Wiener filter is a simple approach to overcome this problem. Rather than dividing by H, the Wiener filter (also known as the restoration filter) creates an estimate of X as follows:

$$\hat{X}(k_u, k_v) = \frac{Y(k_u, k_v)}{H(k_u, k_v)} \frac{|H(k_u, k_v)|^2}{|H(k_u, k_v)|^2 + \gamma},$$

where $\gamma > 0$. An estimate of x is then computed according to $\hat{x} = \mathcal{F}^{-1}\{\hat{X}\}$.

A blurry image can be found at the following link. This image was created by cross-correlating the source image with a 15×15 box car filter.

`www.stanford.edu/~ndwork/teaching/1706ee102a/hmwk/data/bacBlurred.png`

a) Use a Wiener filter to deblur the image.
b) Create several different deblurred images with different values of γ. Explain your results.

Note: for this problem, do not use any of Matlab's image deblurring algorithms.

Problem 6. *The Discrete Cosine Transform and JPEG Compression*
a) Suppose you are given a real vector of length N as shown below. Suppose you were to store the $DFT\{x\}$ in a computer with floating point precision. How would you store this vector? How many bytes would you need? (Recall: a float requires four bytes.)

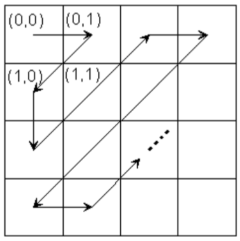
$$x = \begin{bmatrix} x_0 & x_2 & \cdots & x_{N-1} \end{bmatrix}$$

b) Rather than computing the DFT of x, we can compute the DFT of a different vector $\tilde{x}$ defined as follows:

$$\tilde{x} = \begin{bmatrix} x_0 & x_1 & x_2 & \cdots & x_{N-1} & x_{N-1} & \cdots & x_2 & x_1 & x_0 \end{bmatrix}.$$

Note that $\tilde{x}$ is an even vector (mod $2N$). Due to the symmetry of the Discrete Fourier Transform, all the imaginary values of DFT$\{\tilde{x}\}$ are 0. Additionally, DFT$\{\tilde{x}\}$ is even, so we only need to store half of the values in the computer. After taking advantage of this symmetry, how would you store this vector in memory? How many bytes do you need to store it with floating point precision?
What you have just discovered is called the Discrete Cosine Transform (DCT). It's inverse is called the Inverse Discrete Cosine Transform. Matlab has these routines implemented as `dct` and `idct`. Similarly, it has two dimensional implementations: `dct2` and `idct2`.
c) JPEG compression is heavily based on the DCT. An 8-bit image (with values from 0 to 255) has 128 subtracted from each pixel so that values range form -128 to 127. The image is then broken up in to 8×8 blocks and the DCT is calculated on each block. A threshold is applied to the result so that any small values are set to 0. This thresholded data is stored on your computer (with the `*.jpg` extension). When the data is stored on disk, rather than storing all the 0s, we store '0#',

where the number following the 0 says how many zeros there are in a row. This is a variation of *run-length encoding*, and this is where the compression takes place. (There is also a quantization step that we won't discuss in this problem.) The order of the run is shown in the image below.



To reconstruct the image, one unpacks the run into a 2D array of numbers (with a lot of 0s) and applies the inverse 2D DCT.
The value of the threshold determines the quality of the resulting reconstruction. Since some information is lost when a threshold is applied, jpeg is an example of *lossy compression*.

- Download any image that you like. Convert the image to grayscale using `rgb2gray`.
- Compute the DCT of each 8x8 block. (If the size of the image isn't divisible by 8, pad the image with zeros so that it is.)
- Show the original image along side the result of the DCT of each block. Given this, why do you think the run length ordering is as shown above?
- Threshold the result so that approximately 25% of the DCT values become 0. Apply the inverse DCT to each block. Display the thresholded DCT coefficients along side the reconstructed image.
- Threshold the result so that approximately 50% of the DCT values become 0. Apply the inverse DCT to each block. Display the thresholded DCT coefficients along side the reconstructed image.

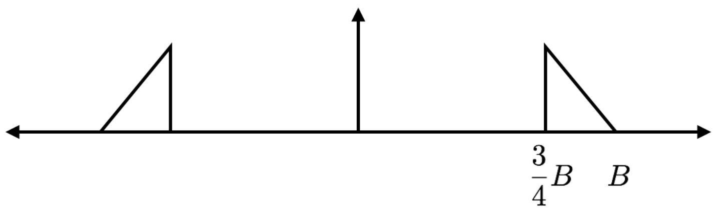
You've compressed data!
d) What was the upside of using the DCT over the DFT? What's the main downside of using the DCT over the DFT?

Problem 7. *A Non-linear System*
Consider the following system:

$$S\{x(t)\}(\gamma) = \int_{-\infty}^{\gamma} x(\tau)d\tau + 8.$$

a) Is this system linear/non-linear, shift invariant or shift variant, causal or non-causal, memoryless or has memory?
b) Find an expression for $\mathcal{F}\{y\}$ where $y = S\{x\}$ for any arbitrary x given the system's impulse response h.

Problem 8. Consider the spectrum of a real signal with bandwidth B shown in the figure below.



The Nyquist theorem states that the signal can be perfectly reconstructed using sinc interpolation when the sampling frequency is greater than twice the bandwidth. In this case, however, we can use some tricks and get away with any sampling rate just greater than $B/2$. Explain how.

Problem 9. *Non-integer Shifts of Sequential Data*
a) Suppose you are given a sequence x. Determine an algorithm to shift the sequence by a non-integer shift.
b) Shift the following sequence by 0.1, 0.2 and 0.5 samples:

$$x[n] = (\ldots, 0, 0, 1, 8, \underline{12}, 7, 5, 0, 0, \ldots).$$

Note that the underline denotes the 0 index value of the sequence. Plot your results using `stem` in Matlab.

Problem 10. Suppose you have two linear FIR filters with impulse responses h_1 and h_2, with corresponding transfer functions H_1 and H_2. Draw the block diagram of a system with transfer function $H = H_1 H_2$.

Problem 11. Consider a function $x_c : \mathbb{R} \to \mathbb{C}$ and a sequence $x : \mathbb{Z} \to \mathbb{C}$ such that $x[n] = x_c(nT)$ for some sampling period $T \in \mathbb{R}$. Determine the relationship between $\mathcal{F}\{x_C\}$ and $\mathcal{D}\{x\}$.

Problem 12. *Alternate Discrete-Time Fourier Transform*
The definition of the Discrete-Time Fourier Transform that we've been using in this class is

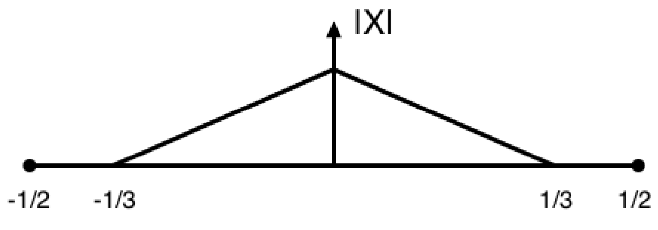
$$X(s) = \mathcal{D}\{x\}(s) = \sum_{n=-\infty}^{\infty} x[n] \exp\left(-i\, 2\pi s\, n\right).$$

Another popular definition of the Discrete-Time Fourier Transform is

$$\hat{X}(\omega) = \hat{\mathcal{D}}\{x\}(\omega) = \sum_{n=-\infty}^{\infty} x[n] \exp\left(i \omega n\right).$$

(Note the sign of the argument of the exponential.) Derive a formula for the inverse of $\hat{\mathcal{D}}$.

Problem 13. Consider a sequence x with spectrum X. The magnitude of the spectrum is shown in the figure below.



Draw the magnitude of the spectrum after downsampling by 2 and then upsampling by 2.