EECS 16B Designing Information Devices and Systems II
Spring 2019 UC Berkeley

Homework 13

This homework is due on Wednesday, May 08, 2019, at 11:59PM.
Self-grades are due on Saturday, May 11, 2019, at 11:59PM.

1. Conjugate symmetry

The DFT basis is naturally complex. However, many signals that we are interested in understanding are real-valued. It is natural to wonder if anything special happens to real-vectors viewed in the DFT basis.

Why are such questions natural? Because a complex number can be viewed as a pair of real numbers (the real component and the imaginary component). So, when we take n real numbers and transform them into n complex numbers, it is natural to wonder where the extra n real numbers have come from? Here we will see how the DFT exhibits something that we can call "conjugate symmetry" and that really, there are only n real numbers that determine everything.

(a) **Convert $\vec{h} = [1,2,1,0]^T$ to the DFT basis by computing $U\vec{h}$**, where U is the DFT basis with $n = 4$. **Plot the components using both Cartesian and Polar coordinates.** Do you see something?

(b) **Do the same for $\vec{h} = [2,1,-1]^T$ by computing $U\vec{h}$**, where U is now the DFT basis with $n = 3$. Do you see something?

(c) Let $\vec{x}$ be a real vector of length n, and let $\vec{X} = U\vec{x}$ be $\vec{x}$ in the DFT basis. **Show that the k-th component of $\vec{X}$ satisfies $X[k] = (X[n-k])^*$, for $0 \leq k \leq \lfloor \frac{n}{2} \rfloor$. Check that this holds in parts (a) and (b).**

(d) **Show that $X[0]$ is real if $\vec{x}$ is real. Check that this holds in parts (a) and (b).**

(e) If n is *even*, **show that $X[\frac{n}{2}]$ is real if $\vec{x}$ is real. Check that this holds in part (a).**

2. Two-dimensional DFT

One common way to use the DFT is to extend it to two-dimensional signals, such as images. We often refer to an image as being in the "image domain" and its 2D DFT to be in the "frequency domain". We can model a 2D image as an $N \times N$ array m_f where

$$m_f = \begin{bmatrix} m_f[0,0] & m_f[0,1] & \cdots & m_f[0,N-1] \\ m_f[1,0] & m_f[1,1] & \cdots & m_f[1,N-1] \\ \vdots & \vdots & \cdots & \vdots \\ m_f[N-1,0] & m_f[N-1,1] & \cdots & m_f[N-1,N-1] \end{bmatrix} \tag{1}$$

Let the matrix M_F be the 2D DFT coefficients of m_f. The 2D DFT coefficients are given by

$$M_F[u,v] = \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} m_f[x,y] e^{-i\frac{2\pi}{N}(ux+vy)} \tag{2}$$

Conversely, an image-domain signal can be expressed by the linear combinations of its 2D DFT coefficients.

$$m_f[x,y] = \frac{1}{N^2} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} M_F[u,v] e^{i\frac{2\pi}{N}(ux+vy)} \tag{3}$$

In this problem, we will play with images with 2D DFT using the IPython notebook. Use the provided IPython notebook file accompanying this homework.

(a) The equation (2) is the result of applying the DFT to the column vectors of m_f first (let the result be $m_{f,c}$), and then applying DFT again to the row vectors of $m_{f,c}$ to get M_F. Demonstrate this using the IPython notebook with the image `campus_256.jpg`. **What is the numeric normed difference between the 2D DFT calculated via the 2D DFT function `np.fft.fft2()` and via computing the 1D DFT on the columns and then the rows of the image?**

(b) The equation (3) is the result of doing inverse DFT on the row vectors of M_F first (let the result be $M_{F,r}$), and then applying inverse DFT again to the column vectors of $M_{F,r}$. Demonstrate this fact using the IPython notebook with the image `campus_256.jpg`. **What is the numeric normed difference between the original image and the reconstructed image via the 2D DFT function?**

(c) We can modify the 2D DFT coefficients of an image by zeroing out its high- or low-frequency coefficients. Use the IPython notebook to demonstrate the effects of modifying the 2D DFT coefficients of the image `campus_256.jpg`. **Describe qualitatively what you view about the low-pass and high-pass versions of the image. What can you see in each image?**

(d) Look at the provided image `einstein_monroe_256.jpg` and **comment about what is special of this image. What does it look like if you look at it with your peripheral vision (or from very far away)? What does it look like when you are focusing on it?** Use the IPython notebook to separate the hybrid image into one with a low-frequency Marilyn Monroe and one with high-frequency Albert Einstein. **What value of r did you use to separate the two images best?**

(e) Look at the provided image `mona_lisa_.jpg`. **Comment on what you observe about the low-pass and high-pass filtered versions of the Mona Lisa. At what value of r do you see the most interesting difference?** *(There is no definite "right" value for r here; play around with a few values and see what changes in the images.)*

(f) (Optional, out of scope) **How would you show that the 2D DFT is an orthonormal transformation into a new basis?**

3. Amplitude Modulation (AM) In electronic communication, transmission is achieved by varying *some aspect* of a *higher frequency signal*, ("carrier signal)", with some information-bearing waveform ("base signal") to be sent, such as an audio or video signal, a process known as **"modulation"**. Amplitude modulation is a specific scheme of modulation, where the amplitude of the carrier oscillations is varied. In this question, you will be led through the mixing of some simple base and carrier signals so you know the magic behind your traditional AM radio.

Recap of DFT: Consider a sampled signal that is a function of discrete time $x[t]$. We can represent it as a vector of discrete samples over time $\vec{x}$, of length n.

$$\vec{x} = \begin{bmatrix} x[0] & \dots & x[n-1] \end{bmatrix}^T \tag{4}$$

Let $\vec{X} = \begin{bmatrix} X[0] & \dots & X[n-1] \end{bmatrix}^T$ be the signal $\vec{x}$ represented in the frequency domain, that is

$$\vec{X} = U^{-1}\vec{x} = U^*\vec{x} \tag{5}$$

where U is a matrix of the DFT basis vectors.

$$U = \begin{bmatrix} | & & | \\ \vec{u}_0 & \cdots & \vec{u}_{n-1} \\ | & & | \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & e^{j\frac{2\pi}{n}} & e^{j\frac{2\pi(2)}{n}} & \cdots & e^{j\frac{2\pi(n-1)}{n}} \\ 1 & e^{j\frac{2\pi(2)}{n}} & e^{j\frac{2\pi(4)}{n}} & \cdots & e^{j\frac{2\pi 2(n-1)}{n}} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & e^{j\frac{2\pi(n-1)}{n}} & e^{j\frac{2\pi 2(n-1)}{n}} & \cdots & e^{j\frac{2\pi(n-1)(n-1)}{n}} \end{bmatrix} \tag{6}$$

Alternatively, we have that $\vec{x} = \frac{1}{n}U\vec{X}$ or more explicitly

$$\vec{x} = X[0]\vec{u}_0 + \cdots + X[n-1]\vec{u}_{n-1} \tag{7}$$

In other words, $\vec{x}$ *can be always represented as* a linear combination of the DFT basis signals $\vec{u}_m$ with coefficients $X[m]$.

(a) Let n be the number of samples in a vector. Suppose we have a signal that is a complex exponential, $s[t] = e^{j\frac{2\pi Kt}{n}}$, which has a frequency of $\frac{K}{n}$ for some integer K and the number of samples n. In vector form it is given by:

$$e^{j\frac{2\pi Kt}{n}} \Rightarrow \begin{bmatrix} 1 & e^{j\frac{2\pi K}{n}} & e^{j\frac{2\pi K(2)}{n}} & \cdots & e^{j\frac{2\pi K(n-1)}{n}} \end{bmatrix}^T = \vec{u}_K \tag{8}$$

Note, that this is the Kth DFT basis vector.

Mixing two signals is defined to be *the process of element-wise multiplying them together*, i.e. computing $e^{j\frac{2\pi Kt}{n}}x[t]$. This amounts to *element-wise multiplication* $\odot$:

$$e^{j\frac{2\pi Kt}{n}}x[t] = (\vec{u}_K \odot \vec{x})[t] \tag{9}$$

The idea is, instead of transmitting the signal $\vec{x}$ directly, we would transmit the mixed signal. The mixing process is known as modulation. There are two questions you probably have are: how can we demodulate a signal? and, what does amplitude modulation do for us? You will know both answers by the end of this problem. But you don't have to answer them now.

Let's start simple. Suppose our signal $x(t)$ is a simple complex exponential with frequency $\frac{k}{n}$, that is $\vec{x} = \vec{u}_k$.

What is $\vec{x}$ written in the frequency domain? Now mix $\vec{x}$ with the complex signal $\vec{u}_K$. What is $\vec{u}_K \odot \vec{x}$? What is this new signal written in the frequency domain? What does mixing with a complex exponential do to the frequency of the signal?

(b) Now suppose that $\vec{x}$ is a cosine signal, that is $x[t] = \cos\left(2\pi\frac{k}{n}t\right)$. **Mix this new $\vec{x}$ with the complex exponential $\vec{s} = \vec{u}_K$. What is the frequency domain representation of this new signal and how does it differ from $\vec{x}$?**

(c) With the above $x[t] = \cos\left(2\pi\frac{k}{n}t\right)$, let's go one step further by mixing it with a carrier signal $\vec{s}$, where $s[t] = \cos\left(2\pi\frac{K}{n}t\right)$.

This is practically important because complex-exponentials need to be realized in a physical way that is real to build real-world AM radios.

What is the frequency domain representation of this new signal and how does it differ from $\vec{x}$?

Assume here that K is much bigger than k and in-turn, n is much bigger than K.

(d) Now that you see the effect of mixing the base signal with a cosine wave as the carrier signal. It is natural to wonder how can we undo this process. The most obvious answer — divide through by the cosine — suffers from the problem of having to divide by numbers that might be close to zero. Dividing by very small numbers is generally a bad idea in engineering contexts.

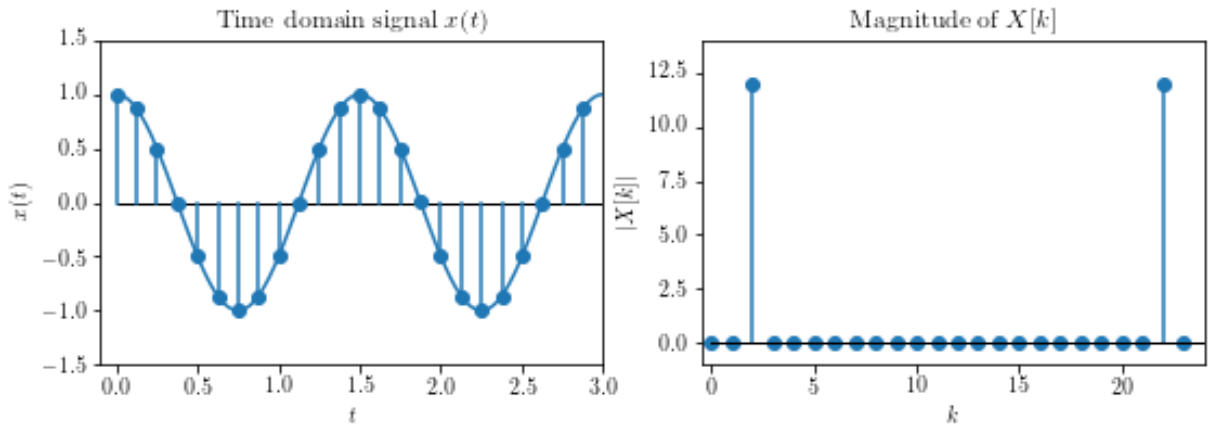
The technique that is used is called **de-modulation**. Now, mix $\cos\left(\frac{2\pi Kt}{n}\right)$ with the signal you get from the previous question. **How many resulting terms do you get in the frequency domain? Considering the fact that $K \gg k$, what are the frequencies relative positions on the frequency spectrum?**

(e) **Have we recovered the original signal? If not, what additional steps would we have to take?** ***(Hint: is there a subspace in which we have correctly recovered the original signal? How can we just keep that subspace?)*** Refer to the iPython notebook for the implementations for mixing and demodulating the signals. Comment on what you see.

(f) Now Assume that we want to send two signals, $x_1[t] = \cos\left(2\pi\frac{k_1}{n}t\right)$ and $x_2[t] = \cos\left(2\pi\frac{k_2}{n}t\right)$. Assume that the carrier frequency $K \gg k_1$ and $K \gg k_2$. **If we modulate the first signal by $s_1[t] = \cos\left(2\pi\frac{K}{n}t\right)$ and the second signal by $s_2[t] = \cos\left(2\pi\frac{2K}{n}t\right)$ and add them together, can we recover both signals?**

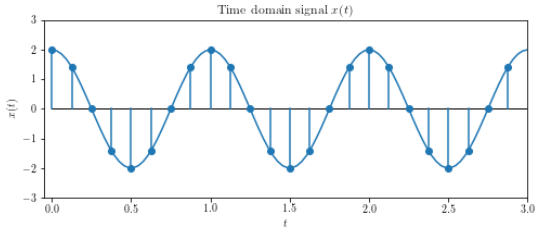
4. DFT Sampling Matching

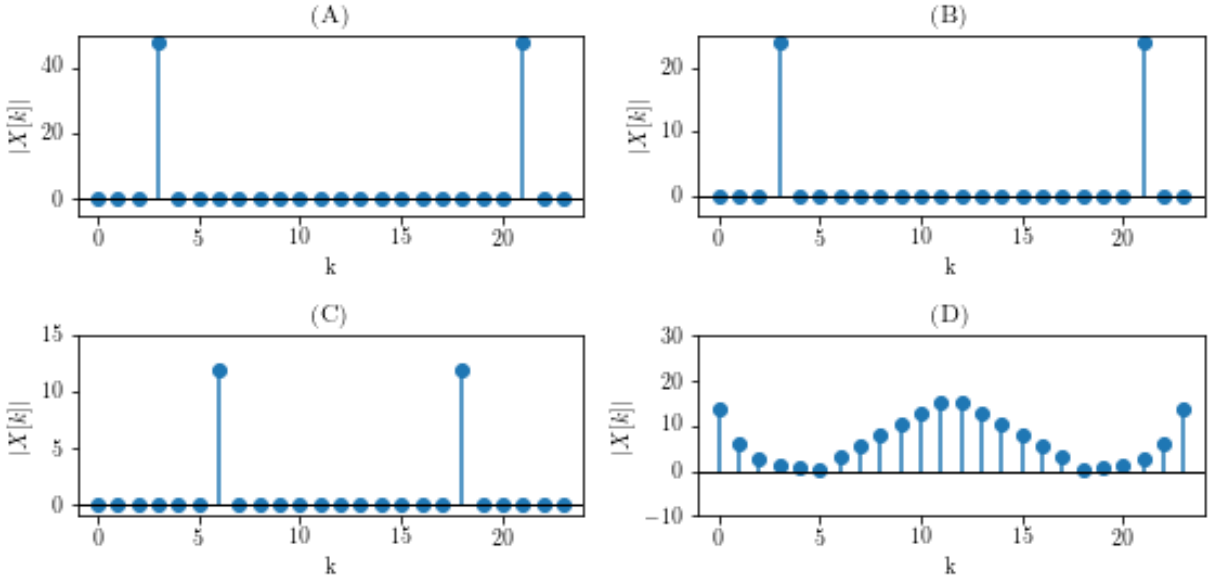
(a) A sampled sinusoidal time domain signal and its DFT coefficients are given below:



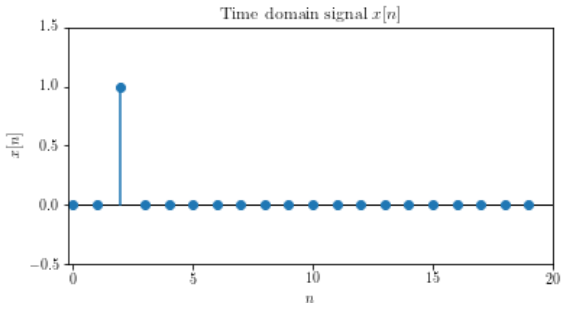
Remember that we saw in discussion that the DFT of a pure harmonic of the form $x \mapsto A_l \cos(2\pi l f_0 + \theta_l)$ has a very particular structure, and that its non-zero coefficients are related to A_l and θ_l.

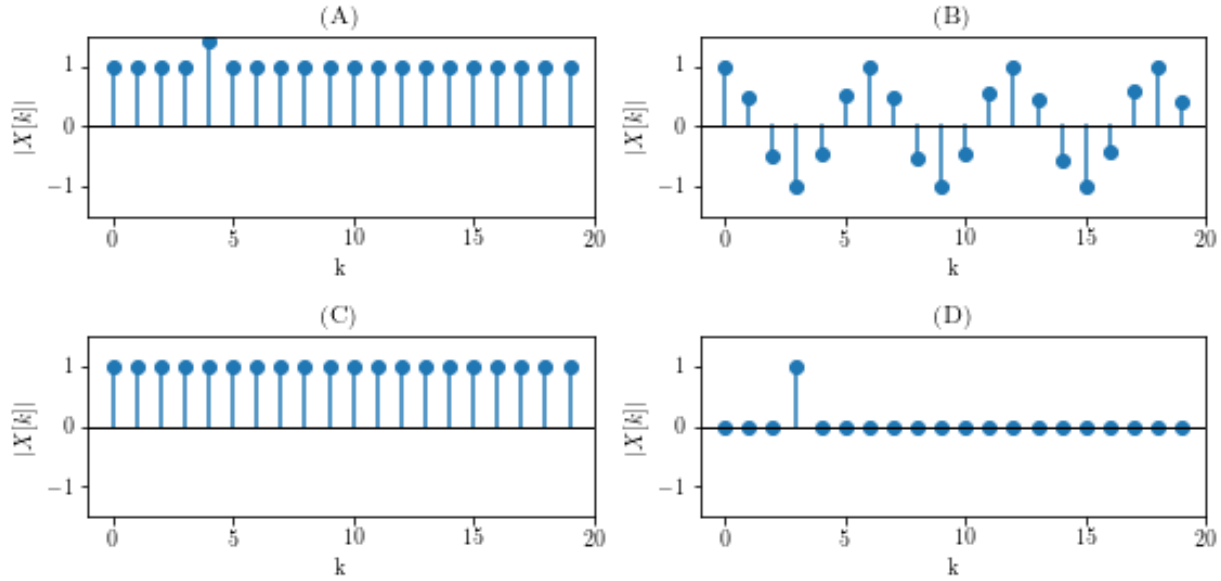
Now given the following sinusoidal time domain signal, **which of the options below shows the correct DFT coefficient magnitudes?**



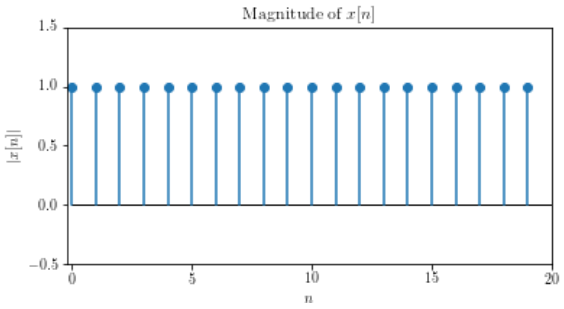


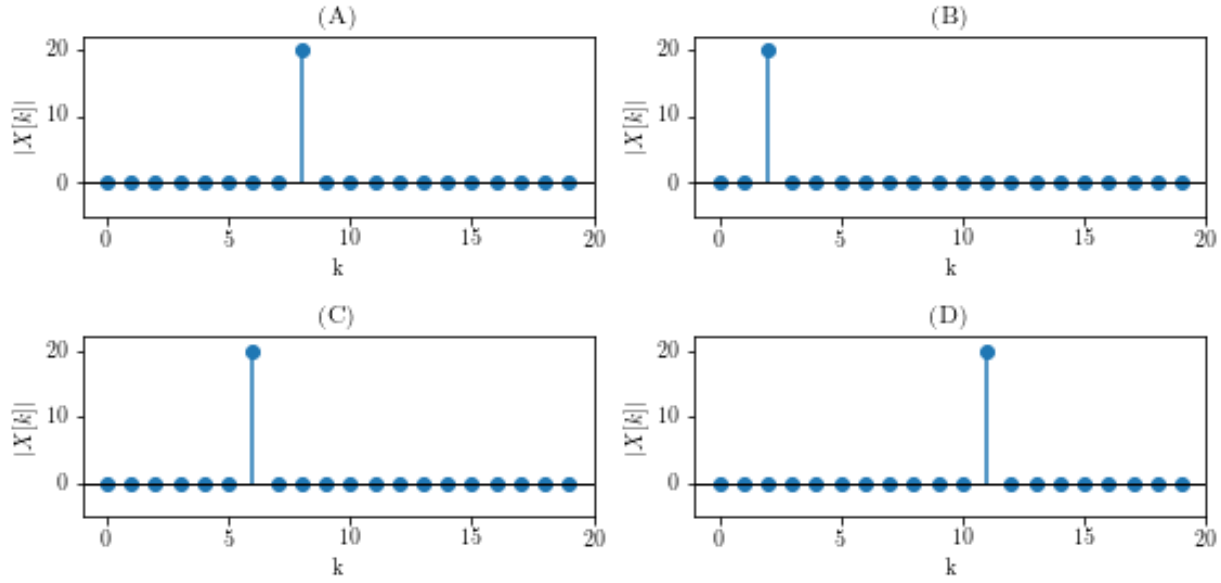
(b) Given the time domain signal below, **which of the options below shows the correct DFT coefficient magnitudes?**





(c) Given the magnitude of the (complex) time domain signal below, **which of the options below shows the correct DFT coefficient magnitudes?** *Hint: this is a trick question!*





5. Denoising signals using the DFT

Professor Maharbiz is sad. He just managed to create a beautiful audio clip consisting of a couple pure tones with beats and he wants Professor Lustig to listen to it. He calls Professor Lustig on a noisy phone and plays the message through the phone. Professor Lustig then tells him that the audio is very noisy and that he is unable to truly appreciate the music. Unfortunately, Professor Maharbiz has no other means of letting Professor Lustig listen to the message. Luckily, they have you! You propose to implement a denoiser at Professor Lustig's end.

(a) In the IPython notebook, listen to the noisy message. **Plot the time signal and comment on visible structure, if any**.

(b) Take the DFT of the signal and plot the magnitude. In a few sentences, **describe what the spikes you see in the spectrum are**. *Hint: take a look at the documentation for numpy.fft.fft.*

(c) There is a simple method to denoise this signal: Simply threshold in the DFT domain! **Threshold the DFT spectrum by keeping the coefficients whose absolute values lie above a certain value. Then take the inverse DFT and listen to the audio.** You will be given a range of possible values to test. Write the threshold value you think works best.

Yay, Professor Maharbiz is no longer sad!

6. Non-linear Transforms and Harmonics

In more advanced classes, like EE120, you will learn that passing an input through a linear system will not add new frequency components in the output. But, in this question we will look at a simple non-linear transform $f_p(x) = x^p$, and understand the harmonics present in the output.

(a) Beginning with something simple, let our input $x(t) = \cos(2\pi f_0 t)$, where $f_0 = 1$Hz. We also have our output as $y_1(t) = f_2(x) = x^2$. **Find the harmonics present in the output given $x(t)$. Furthermore, by hand, sketch $y_1(t)$ (please refrain from using a plotting software/graphing calculator).** *Hint: Consider writing* $\cos(x)$ *as a sum of two exponentials.*

(b) Next, let our output be given by $y_2(t) = f_p(x) = x^p$. **Find the harmonics with the input given in part (a).**

Hint: Consider the binomial theorem [1]*:*

$$(x+\frac{1}{x})^p = x^p + px^{p-1}\frac{1}{x} + \frac{(p)(p-1)}{2}x^{p-2}\frac{1}{x^2} + \frac{(p)(p-1)(p-2)}{2\cdot 3}x^{p-3}\frac{1}{x^3} + \cdots \frac{(p)\cdots(2)}{2\cdot 3\cdots(p-1)}x\frac{1}{x^{p-1}} + \frac{1}{x^p}$$
$$= \sum_{k=0}^{p} \frac{p!}{k!(p-k)!}x^{p-k}\frac{1}{x^k}$$

(c) Now, let's go a step further and consider the following input,

$$x(t) = \cos(2\pi f_1 t) + \cos(2\pi f_2 t).$$

Here, $f_1 = 1$ MHz $= 10^6$ Hz and $f_2 = 10^6 + 1$ Hz. Let's again consider the output to be given as $y_3(t) = f_2(x) = x^2$. **Find the harmonic with the minimum frequency.**

[1] For further reading, please see here

7. Write Your Own Question And Provide a Thorough Solution.

Writing your own problems is a very important way to really learn material. The famous "Bloom's Taxonomy" that lists the levels of learning is: Remember, Understand, Apply, Analyze, Evaluate, and Create. Using what you know to create is the top level. We rarely ask you any homework questions about the lowest level of straight-up remembering, expecting you to be able to do that yourself (e.g. making flashcards). But we don't want the same to be true about the highest level. As a practical matter, having some practice at trying to create problems helps you study for exams much better than simply counting on solving existing practice problems. This is because thinking about how to create an interesting problem forces you to really look at the material from the perspective of those who are going to create the exams. Besides, this is fun. If you want to make a boring problem, go ahead. That is your prerogative. But it is more fun to really engage with the material, discover something interesting, and then come up with a problem that walks others down a journey that lets them share your discovery. You don't have to achieve this every week. But unless you try every week, it probably won't ever happen.

8. Homework Process and Study Group

Citing sources and collaborators are an important part of life, including being a student! We also want to understand what resources you find helpful and how much time homework is taking, so we can change things in the future if possible.

(a) **What sources (if any) did you use as you worked through the homework?**

(b) **Who did you work on this homework with?** List names and student ID's. (In case of homework party, you can also just describe the group.)

(c) **How did you work on this homework?** (For example, *I first worked by myself for 2 hours, but got stuck on problem 3, so I went to office hours. Then I went to homework party for a few hours, where I finished the homework.*)

(d) **Roughly how many total hours did you work on this homework?**

Contributors:

- Yen-Sheng Ho.
- Ming Jin.
- Siddharth Iyer.
- Brian Kilberg.
- Aditya Arun.
- Jaijeet Roychowdhury.