# CSC 476/676 Homework 3. Fourier Transform and Pyramid Image Blending

# Due: March 9th, 2021

Total points (50pts) + Bonus (20pts)

## Instructions:

## In this homework, you can use numpy.fft, scipy.fftpack and matplotlib.

Please turn in a Google Colab link/notebook together with your media files.

You must turn in a webpage (.html) as your result (see bottom of the instructions).

## If you choose to do it on your computer, please turn in your .py or .ipynb file and your write up and upload on blackboard.

## This exercise is for each individual student. No copy of code from each other or the internet. This is a short homework that is expected to be done in a few hours.

## Some useful tutorials:

## <https://docs.scipy.org/doc/numpy/reference/routines.fft.html>

<https://docs.scipy.org/doc/scipy/reference/tutorial/fftpack.html>

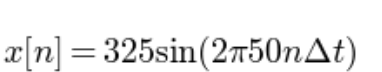
## <https://medium.com/@y1017c121y/python-computer-vision-tutorials-image-fourier-transform-part-2-ec9803e63993>

## No late work is accepted. Read syllabus for late policy.

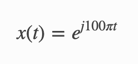
1. (5pts) Plot both continuous and discrete signals of the function



And also the discrete signals



Where n is sampled from 0 to 50.

1. (5pts) Consider a complex-valued signal,  Can you plot the real and imaginary part

Of the function when t have a range of (-0.02, 0.05)?

Hint: np.exp(2j\*np.pi\*50\*t).real. would return real parts

np.exp(2j\*np.pi\*50\*t).imag. would return imaginary parts

1. (5pts) In this exercise, we want to generate the same plot as Slide No. 20 in the lecture9 slides (Figure 1.20 in Chapter SignalProcessing.pdf).

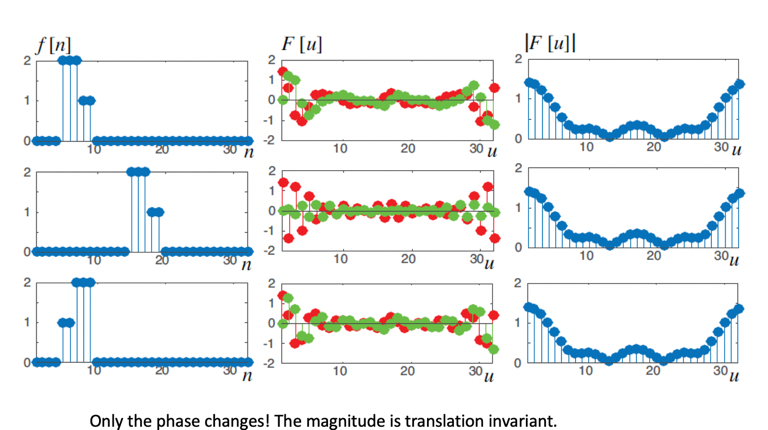
Take a 1D discrete signal, eg. f[n], show that the foureier transform

Of a translated signal f[n-x] and its left-right mirror signal would have resulted in the same magnitude. Basically, use numpy.fft and matplotlib, generate the following graphs. You can use a different f[n], use N= 32 as the Fourier coefficients.

Hint: you can use similar code to create discrete signals in numpy:

**>>>** t = np.linspace(0,20,N)

**>>>** x = np.exp(-t/3)\*np.cos(2\*t)

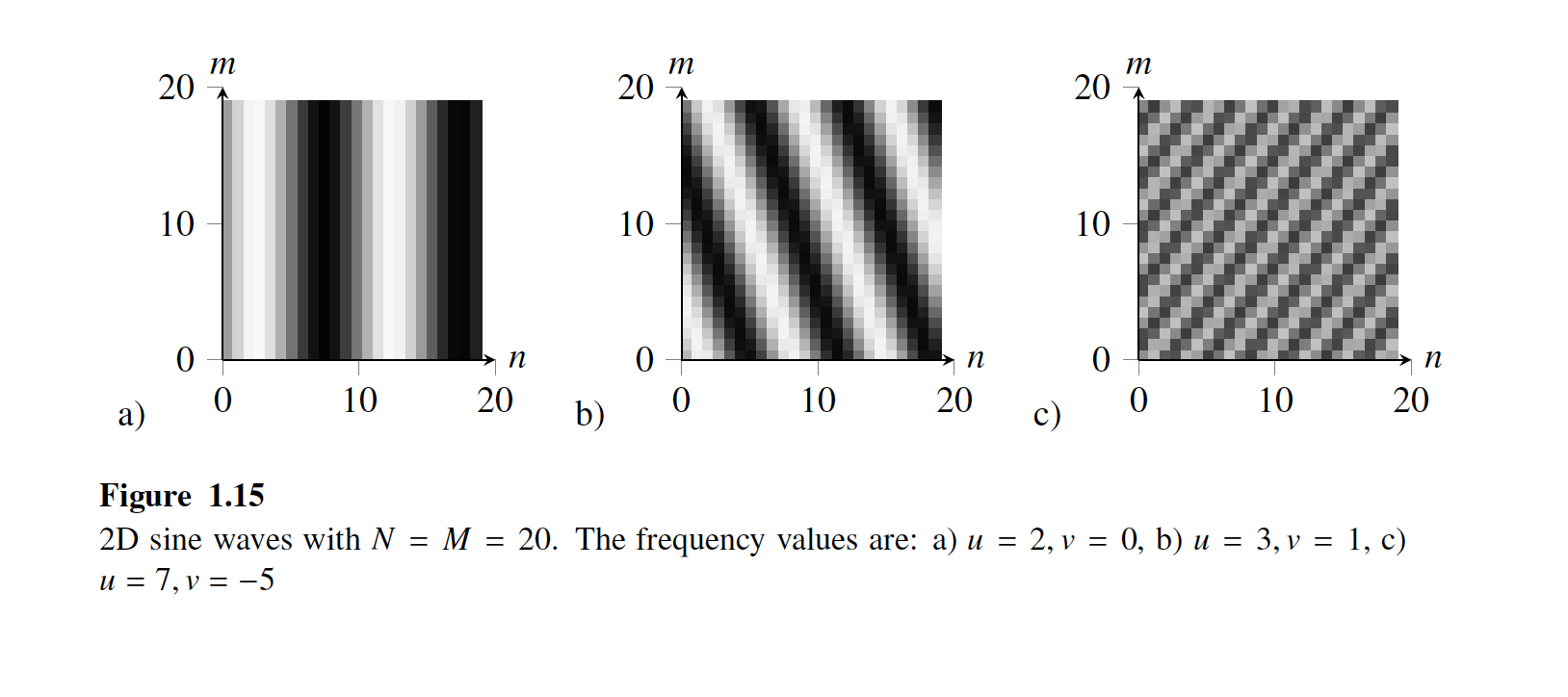


The red and green dots ae real and imaginary parts of the FFT of the signal.

1. (5pts) Plot the fourier transform of the following function for frequency below 50hz, using the numpy.fft



1. (5pts) Create the following 2D sine wave grating patterns as images and show their 2D Fourier transforms of the three images. You must plot both thee input images in real domain and the Fourier domain.



1. (25pts) Pyramid Blending: Write a program that takes an input two color images and a binary mask images and produces the Laplacian pyramid blend of the two images.

* Construct the Laplacian pyramid for each image.
* Construct the Gaussian pyramid for the two mask images (the input image and its complement)
* Multiply each Laplacian image by its corresponding mask and sum the images
* Reconstruct the final image from the Blended Laplacian pyramid.

Here are some more detailed instructions:

*Laplacian Pyramids.*  Implement functions that build and collapse Laplacian pyramids. The input to the Laplacian pyramid building function is an image and the output is both the Gaussian and Laplacian pyramids for the image. The situation is reversed for collapsing a Laplacian pyramid (but really all that is needed is the lowest level Gaussian along with all levels of the Laplacian pyramid). You can use the *interpolate* and *decimate*functions discussed in class, or you can implement your own. Note that building and collapsing a Laplacian pyramid should yield the original input image. Before you do any blending, checking whether your code can successfully collapsing into multiple sub-bands and reconstruct an image.

Create an image that shows multiple levels of the Gaussian and Laplacian pyramid of an image and include this in your write up along with an explanation of image pyramids in your own words.

*Laplacian Blending.* Implement Laplacian image blending for a pair of images and a binary mask. In Laplacian blending, a new Laplacian pyramid is built by alpha blending each corresponding Laplacian pyramid level for the input image pairs using a Gaussian pyramid of the binary mask as the alpha mask. The lowest level of the new pyramid also includes the alpha blend of the lowest level Gaussian images in the two pyramids (you can blend all levels of the Gaussian pyramids if you want, but you technically only need the coarsest level). The resulting Laplacian pyramid is collapsed to recover the blended image.

Try your multi-scale Laplacian blending method out on several sets of images. Show *at least three* results you consider "good" and one you consider "poor" in your write up. For each image, discuss why you think this approach did or did not work well.

You can compare your results with a simple “copy and paste” methods by directly apply mask on one image and blend with another

## Tips

To blend a pair of images you need a source image, target image, and mask. The source image contains the data you are inserting into the target image. The mask indicates where in the output image the data comes from the source image. It helps if all three images are the same size. A simple mask can be built by dividing the image plane in half. This will blend two images together along the middle seam, which is what is done for the famous apple/orange hybrid.

More advanced masks can be created using an image editing program that uses layers, such as [Gimp](http://www.gimp.org/). A simple procedure is:

* Open the target image
* Open the source image
* Select a region in the source image (rectangle, oval, lasso). Copy it.
* Paste the region into the target image as a new layer.
* Move or rescale the new layer so it appears where you want it to above the target layer.
* Create a new transparent layer.
* Paint in this layer with white paint where you want the source image data to appear in the final image.
* For each layer in turn, make the layer exclusively visible, select Save As... and save the file with a unique name, like myblend-source.png.

Note that Gimp on the Mac is a little annoying because you have to click in a window to change focus before you can interact with the window.

Valid mask values used for alpha blending are between zero and one.  
=aS+(1-a)T  
If you load an alpha mask from a file, you will probably need to modify the image data to fit within this range. You may also want to make sure you are using the same mask for all color channels.

## Bells & Whistles (Extra Points)

* (5pts) Be exceptional creative of acquiring and your choices of source and target images.
* (5pts) You can discuss the shortcomings you discovered with the Pyramid method. When does the method fail?
* (5pts) Implement the two-band Laplacian blending described in [Brown and Lowe, Recognizing Panaroams, ICCV 2003](http://matthewalunbrown.com/papers/iccv2003.pdf). You can simply jumpt to section 5, last two paragraphs. Compare with the multi-band method you have already have implemented.
* (5pts) Implement Poisson image blending (Perez03). See attached paper.

## Write up

For this project, and all other projects for the rest of the semester, you must do a project report in HTML. We provide you with a placeholder .html document, which you can edit. In the report you will describe your algorithm and any decisions you made to write your algorithm a particular way. Then you will show and discuss the results of your algorithm. In the case of this project, show the results of your blending algorithm and show the good and bad examples. Also, discuss anything extra you did. Feel free to add any other information you feel is relevant. If you performed this assignment as a team, please indicate this in the start of your webpage and credit each person for his/her contributions.

Please do not move the highlighting folder that contains important CSS sheets (unless you want to do your own web design). Just edit the .html file and keep the folder structure when you submit your code.

Please upload your code and website onto GitHub.

## Handing in

This is very important, as you will lose points if you do not follow instructions. Every time after the first that you do not follow instructions, you will lose 5 points. The folder you hand in must contain the following:

* README - text file containing anything about the project that you want to tell the TAs
* code/ - directory containing all your code for this assignment
* html/ - directory containing all your html report for this assignment (including images). Only this folder will be published to the course web page, so your webpage cannot contain pointers to images in other folders of your hand-in.
* html/index.html - home page for your results
* Zip the folder and submit to Canvas