CS-GY 6313 / CUSP-GX 6006: Data Visualization
Spring ’26

Homework #1

This homework is due two weeks after the release of this assignment: March 3, 2026.

This homework consists of 2 required questions (15 points) and 2 bonus questions (1 point each). Submit your code as a `.zip` *file together with your code and any other submittable items, such as figures or writeups. You are permitted to use external Python libraries for your work.*

Adhere to NYU’s Academic Integrity rules. While collaboration between students is allowed, your answers and implementation must be your own. If you relied on external resources, websites, publicly-available code, etc. to answer any part of this assignment, you are expected to add an attribution to the original source material (ex. a comment, or a URL).

Part A: Matrix Transformations (5 points):

We have a point rendered (3,4). We have to rotate this around the origin by 30 degrees counterclockwise, then translate it 3 units rightward and 2 units downward. How can we do this using matrix transformation and homogeneous coordinates?

Task: Transform this point based on the instructions above. Use a singular matrix transformation for this task. Report the new coordinates.

Grading Metric: We’ll run the code to check if the matrix transformation was correct and was performed with one matrix transformation.

Part B: Color Map (10 points):

In this part, you will implement a visualization of topological data. We provide starter code for reading and plotting topological data (`datasets/height.csv`) as a scatter plot. The data is formatted, by column, as (latitude, longitude, elevation). This data is referenced in the code as `x`, `y`, and `z`, respectively.

The matplotlib plotter uses a `color_map` function to map each coordinate to a color. The `color_map` function must return an array of RGB values equal to the size of `z`. This array maps each elevation value in `z` to a color. In its current state, however, it simply returns black for all elevations, producing the following result:

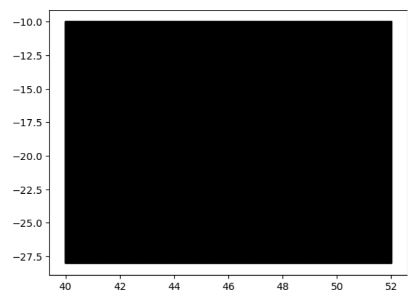
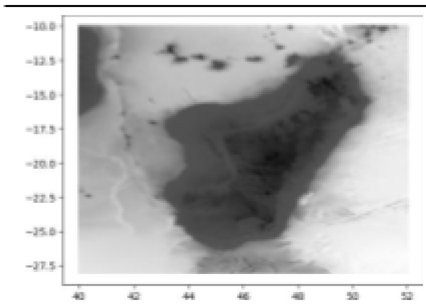
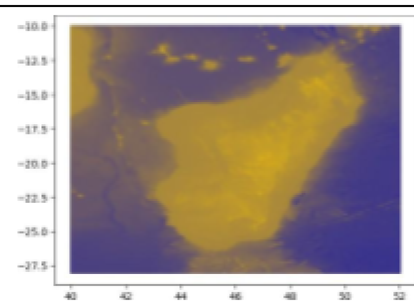


Figure 1: Black Screen

Task: Modify the function `color_map` so that it returns an array of RGB colors that correspond to each elevation value in `z`. The output of the `color_map` function is used to visualize height using a color scheme. Two examples are provided below. Your choice of color scheme does not need to reflect the exact color schemes provided in the example.

Grayscale Example	RGB Example
	

Grading Metric: We will run your code and observe if it produces a similar result to that provided in the examples. If you've chosen a different color scheme than shown in the examples, the visualization must make it easy to distinguish areas of high and low elevation.

BONUS QUESTION: Tone mapping (1 point):

As you learned in class, tone mapping is a technique that maps luminance values to displayable values (0-255). HDR images are typically stored in special file formats. .hdr and .exr are popular formats which store HDR data, and sample images can be found at https://www.pauldebevec.com/Research/HDR/. For this question, you can use the classic memorial church test image.

Task: First, read the hdr image. HDR pixel values are in linear RGB. Tonemap the image using the Reinhard technique.

Tip: See the slides from class.

Metric: We will run your code and ensure it outputs a correct tone-mapped result. Please include an image of the tone-mapped content as a .png.

BONUS QUESTION: Color Frequency Analysis with Fourier Transforms (1 point):

We received an light wave signal (`datasets/signal.wav`) from a friend of ours.

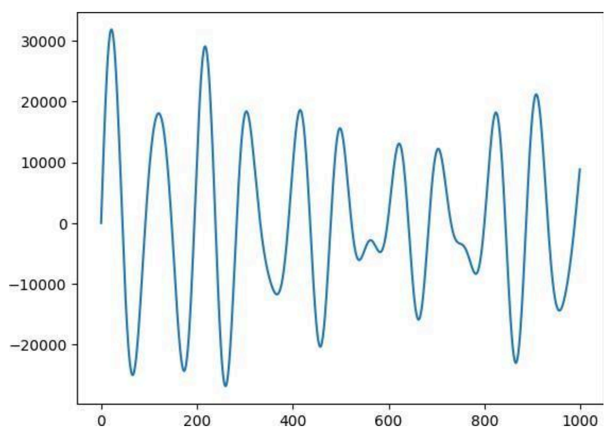


Figure 2: 5 seconds, 44100 Hz

We've been told by this friend that there are 3 colors present in the signal. The signal was sampled at 44100 Hz and lasted for 5 seconds. Unfortunately, it's not obvious just from looking at the raw signal above what the colors are. Can we identify which colors are in the signal?

Task: Modify the `fourier_transform()` function so that it performs a Fourier Transformation on the signal. It must return the `x` and `y` data so that we can plot the transformed signal. Report back on what you believe the frequencies of the three colors are.

Tip: Make sure to limit the x-axis of the transformed data to show only the visible spectrum (400hz to 700hz).

Metric: We'll run your code to make sure the Fourier Transformation has been properly done and that the reported color frequencies are correctly identified.