MATH 260 - Python Programming in Math
Image Compression: processing 2D images using the Fast Fourier Transforms (FFT)

<Xuening Peng, xp27@duke.edu>

Fourier Transform and Signal Processing

FFT is a technique that eases the analysis of signal in frequency domain and an algorithm for fast computation of DFT. FFT contributed to fast computation for real time applications, such as audio processing and image processing. If we are given a 2D signal f(x,y) sampled at N discrete points, the DFT of a 2D image is given by

$$F(u,v) = \sum_{u=0}^{M-1}\sum_{v=0}^{N-1} f(x,y)\cdot e^{2\pi i\left(\frac{n}{N}\right)(ux+vy)}$$

$$, for\ x,y = 0,1,2,\ldots,N-1$$

And the Inverse Fourier Transform is given by

$$f(x,y) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} F(u,v)\cdot e^{-2\pi i\left(\frac{n}{N}\right)(ux+vy)}$$

$$, for\ u,v = 0,1,2,\ldots,N-1$$

The cosine and sine terms are expressed as matrix for various values of (u, x) and (v, y) and FFT of the image is computed by subtracting the sine, cosine and their transposes with the block of the image (AnithaT & Vijayalakshmi, 2018).

Code Implementation and Analysis

To compress an image using FFT (see original image in **figure 1.**), we first transform it both row-wise and column-wise to get a transform image by the command *np.fft.fft2()* (see **figure 2**.). Pixels of this 2D transformed image are Fourier coefficients. In one-dimensional signal processing, the 1D signal is represented as a sum of sine and cosine lines. Therefore, in 2D space, each coefficient consists of the Fourier series along with their sine or cosine values with frequency along both vertical and horizontal directions. And the further the pixels are away from the center, the higher frequency of the pixels in the FFT transformed image.

Figure 3 is a cool visualization of the pixel intensity of this image (Brunton & Kutz, 2017). The taller 3D pixel shapes correspond to the brighter pixels in the image. And the pixel intensity of the landscape of the girl's face in either horizonal or vertical direction exactly represents the sum of sine and cosine waves in x and y with different frequencies.

As for nature images (can be seen by human eyes in nature), as the FFT image shows, most of the Fourier coefficients are negligibly small so that we can truncate these coefficients and remove them. Only a small percentage of the coefficients are large, which are adequate to represent the main features of an image. Therefore, in this case, we can keep the 10%, 5%, 1% or 0.2% of the largest Fourier coefficients while threshold and zero out these negligible ones. And lastly, by implement the inverse of Fourier transformation *np.fft.ifft2()*, we can get the original image with very little loss (see **figure 4.**). As we can see, the first three subplots look almost identical to the original one, and the last one keeping 0.2% coefficients only loses some information and looks a little blurred. Despite of this, it is still clearer enough for the machine or computer to identify main characteristics for further image data training or image processing but saves a lot of storage.

Figure 1. Girl

Figure 2. Grayscale girl and FFT

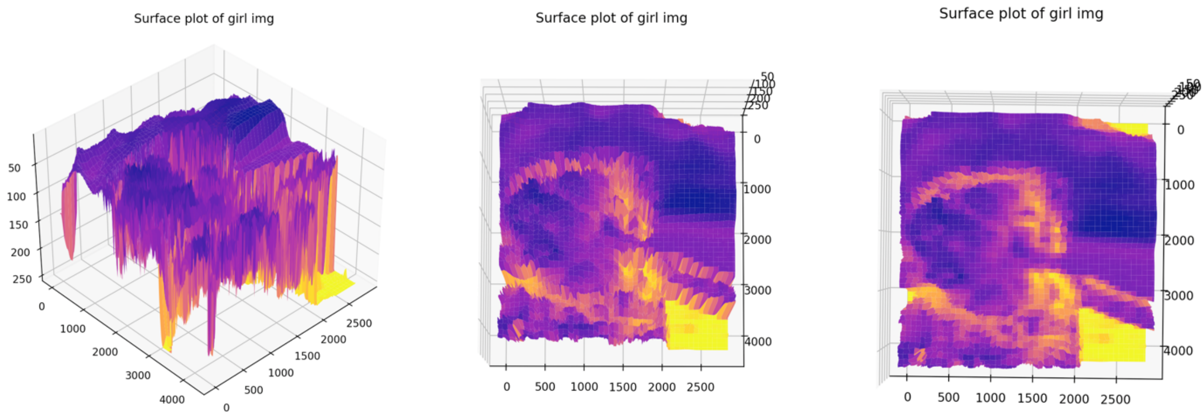

Figure 3. Surface plots

Figure 4. Compressed images with i% high frequency Fourier coefficient kept

Work Cited

AnithaT, G., & Vijayalakshmi, K. (2018). *FFT Based Compression approach for Medical Images*. https://www.semanticscholar.org/paper/FFT-Based-Compression-approach-for-Medical-Images-AnithaT-Vijayalakshmi/9cb3494ad49227bc1ecd1c32660096d1ffa03584

Brunton, S. L., & Kutz, J. N. (2017). *Data Driven Science & Engineering*. 572.