AMATH 482: Home Work 5

Avinash Joshi
bwaseous@uw.edu
AMATH 482 Wi 22
18 March, 2022

Abstract

Signal compression and recovery are incredibly important technologies for modern life given how almost everything relies upon compression, transmission, and decompression of information. This assignment explores how to compress and recover René Magritte's "The Son of Man," Figure 1, using properties of image sparsity and convex optimization. Utilizing the Discrete Cosine Transform (DCT), Figure 2, of the image reveals that many of the coefficients contribute little to the formation of the image - the image is sparse. Using this fact, the image can be reconstructed using a minimal percentage of the DCT coefficients by using the inverse DCT, Figure 3. The image can also be recovered accurately using less than half of the pixels of the figure, Figure 4, using convex optimization because of its sparsity. The recovery using limited observations can then be employed using on an unknown image's pixel observations revealing that the unknown figure, Figure 5, is actually the internet sensation Nyan Cat.

1 Introduction and Overview

The purpose of this assignment is to explore the compressibility of signal, specifically René Magritte's "The Son of Man," and image recovery from limited observations (due to image corruption or compression). The provided image of "The Son of Man" is 292×228 gray-scale pixels but is rescaled to be 2,173, 53×41, pixels for ease of processing. Since this image is comprised of a signal that only has its real part, the 2D Discrete Cosine Transform (DCT) is utilized in order to represent the figure as a sum of cosines with various frequencies for both the rows and columns of the image. The DCT of the vectorized form of the image, DCT(vec(F)) or Dvec(F), is then investigated to confirms its sparsity - the fact that many of its DCT coefficients are near zero. This means that only keeping a portion of the absolute largest coefficients can accurately represent the image and, thus, effectively compresses the image. The compressed Dvec(F) can then be reverted back into the image through the inverse DCT, iDCT or D^{-1}. The percentages of kept coefficients are $5, 10, 20$ and 40%.

The task of signal recover from limited information, however, can also be solved using convex optimization on the DCT of the image. For this task, only a portion of vectorized form of the image, y, are used to recreate the image using a linear relationship between $Ax = y$, where A is an equally small portion taken from the iDCT and x is the approximated DCT of the vectorized image. This small portion of the image is defined by the total number of pixels N, times some decimal rate r which then prescribes the amount of pixels used to reconstruct the image from, M. Solving $Ax = y$ while minimizing $\|x\|_1$ ensures that the sparsity found with the original DCT is preserved. From here, the technique developed using convex optimization on "The Son of Man" can then be used on the randomized portion of the iDCT, B, of an unknown image and its corresponding portion of pixels, y, so that an unknown image can be reconstructed and viewed.

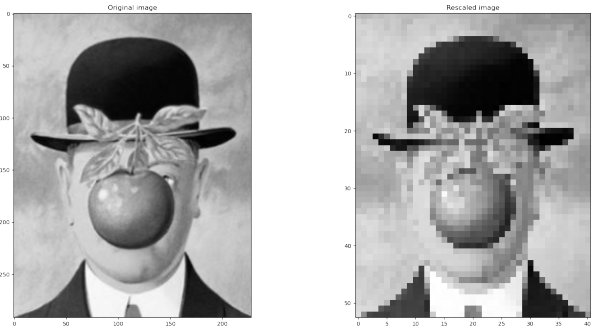

Figure 1: The gray-scaled, original image of René Magritte's "The Son of Man," left, and the rescaled version, right.

2 Theoretical Background

The Discrete Cosine Transform (DCT), or D, expresses discrete data (images, time-series data, etc.) as the summation of a series of cosines with various frequencies k. The DCT is similar to the Fourier Transform but encodes its coefficients as real-valued values. Like most transforms, the DCT has many ways to express itself (8 in fact) and each of them has their own normalizations. One way to express the DCT of a discrete signal $f \in R^K$ is through the orthonormal Type II DCT,

$$\text{DCT}(f)_k = \sqrt{\frac{1}{K}} \left[f_0 \cos \frac{\pi k}{2K} + \sqrt{2} \sum_{j=1}^{K-1} f_j \cos \frac{\pi k (2j+1)}{2K} \right],$$

with its inverse DCT, D^{-1}, defined by the orthonormal Type III DCT, or, more simply, replacing f_k with $\text{DCT}(f)_k$ in the orthonormal Type II DCT. To calculate the DCT for a 2D signal, the DCT is calculated along both dimensions and is simply two DCT's multiplied together. For an image $F \in R^{N_y \times N_x}$, where N_y are the number of rows of pixels and N_x are the number of columns and $N = N_y \times N_x$, with its vectorized version or flattened arrangement of the columns of F, $\text{vec}(F) \in R^N$, D and $D^{-1} \in R^N$ but can be represented by a matrix $D \in R^{N \times N}$. For this assignment, algorithmic discussions consider $D \in R^{N \times N}$, but for theoretical discussions it will be R^N. Constructing D and D^{-1} on the rows and columns of the identity matrix I and doing the Kronecker product between the DCT of the rows and columns ensures that $D^{-1}(D\text{vec}(F)) = \text{vec}(F)$.

It is known that for most signals, especially images, the DCT of the signal is sparse,

or that there are many near-zero or zero-valued coefficients in the image's DCT. It is also known that for most signals only a few coefficients are needed to describe the majority of signal variation. Both of these facts mean that an image can be compressed by only keeping a certain portion of the largest DCT coefficients and then be reconstructed reliably.

An image's sparsity can also be leveraged to do Lasso regression, which is much like other ridge regression tasks, but uses facts associated with the 1-norm penalization, rather than 2-norm penalization, to ensure sparsity. One way to represent the Lasso regression problem is through convex optimization vernacular,

$$\begin{aligned} &\underset{x \in R^{N \times N}}{\text{minimize}} \quad \|x\|_1 \\ &\text{Subject to} \quad Ax = y, \end{aligned}$$

where $A = BD^{-1} \in R^{M \times N}$ with $M < N$ an integer and $B \in R^{M \times N}$ constructed by randomly selecting M rows of the identity matrix I_N. For this assignment, $M = r \times N$ where $0 < r \leq 1$. The observed values of the signal, y, is defined by $y = B\text{vec}(F) \in R^M$. By minimizing the 1-norm of x, the minimizer x^* approximates the $DCT(F)$ by preserving the DCT's sparsity. The 1-norm is defined as the sum of absolute values of the vector x, so by minimizing this, it ensures that many of the values will be close to zero or sparse because the x "travels less distance" in space with the more non-zero coefficients it has. This 1-norm or ℓ-1 regression problem doesn't have a convenient closed-form solution, so numerical methods must be employed. Solving the convex optimization task before ensures that for any B and y, an approximation to the original signal $F \in R^{N_y \times N_x}$ can be reconstructed.

3 Algorithm Implementation and Development

Before any image compression and reconstruction is done, the image is converted into gray scale and compressed into the lower dimension state. This is done through *skimage.io.imread* and *skimage.color.rgb2gray* to read the image then decolor it. To rescale it, *skimage.transform.rescale* is employed with a scaling of 0.18. Using the functions *construct_DCT_MAT* and *construct_iDCT_MAT*, which take in the number of rows and columns of the figure, the generalized D and D^{-1} for any vectorized signal can be computed. This is done using *scipy.fftpack.dct* or *.idct* on *np.eye*[1], for both the rows and columns, and *norm = 'ortho'*. These two DCT's or iDCT's are then multiplied together using *np.kron*, the Kronecker product returning D and D^{-1}.

To explore the sparsity of the $D\text{vec}(F)$, F must be first vectorized by using *.flatten()* and *np.reshape(-1,1)* on the image to turn it into a long column vector. From here, *np.dot*(D, F) is used to define $D\text{vec}(F)$ which is then plotted using *plt.plot*[2] and its mean is calculated using *np.mean*. To find the indices of the largest DCT coefficients, *np.argsort* is called on *np.absDvec*(F) and is then sorted from greatest to least using *[::-1]*. These indices, in conjunction with $D\text{vec}(F)$, D^{-1}, and a percentage value are used in the function *thresholder* which returns the reconstructed signal, F^*, using a certain percentage of

[1] *Numpy* is shortened to *np* for convenience.

[2] *Matplotlib.pyplot* is shortened to *plt*.

the largest DCT coefficients. It does this by calculating the number of coefficients are represented by the percentage value, and runs a for loops through the different sorted indices of $D\text{vec}(F)$ and asks whether the current iteration is greater than the number computed earlier. If the current iteration is greater than that number of coefficients kept, it will set that DCT coefficient to 0. The image is then reconstructed by multiplying the approximated DCT with its inverse and then reshaped and plotted using *plt.imshow*.

From here, the convex optimization part of the assignment is solved using a function called *twosolver* that takes in some decimal value r, the vectorized form of F, and its DCT inverse; then, it returns three separate approximations of the image. It does this by first calculating the number of observations, M, by rounding r by the length of $\text{vec}(F)$. In a *for* loop of 3 iterations, it computes what rows of the I_N to keep using *np.random.choice* with the length of $\text{vec}(F)$ and $size = M$. This vector is then called on the rows of I_N using *np.eye[indices,:]* producing B. y is computed by multiplying B with $\text{vec}(F)$, and likewise is done for A with D^{-1}. x is initialized using *cvx.Variable*[3] with the length $\text{vec}(F)$ with the problem defined as *cvx.Problem* on *cvx.Minimize(cvx.norm1(x))* and the side condition *[A@x == y]*. The problem is then solved by using *problem.solve* using *solver = 'CVXOPT'*. Finally, to compute F^*, the optimal value x^* is extracted and reshaped into a column vector using *x.value* then multiplied into D^{-1} and reshaped. It is then plotted using *plt.imshow*. An almost identical process is used to reconstruct the unknown image as described by *twosolver*, but the unknown B and y are read and extracted using *with np.load as data*. D and D^{-1} are reinitialized using the DCT and iDCT functions but with specified dimensions of 50×50.

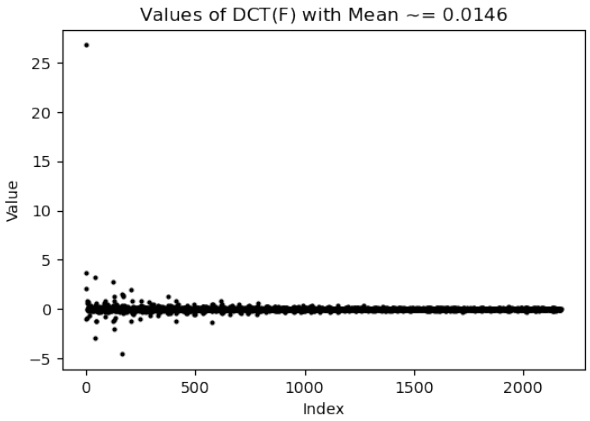

Figure 2: Only a light splattering of coefficients are of a noticeable size while the rest form a thick band around the a value of 0. This visually confirms the sparsity of the DCT coefficients of the image.

4 Computational Results

By following the steps outlined in Section 3 - Algorithm Implementation and Development, the majority of the DCT coefficients are graphically confirmed to be near-zero or zero valued as shown in Figure 2. Because many of the DCT coefficients are near-zero, the image is sparse and thus only keeping a certain portion of the largest DCT coefficients is an appropriate measure to compress the image. The reconstruction of "The Son of Man" using $5, 10, 20, 40\%$ of the coefficients is then shown in Figure 3.

[3] *cvxpy* is shortened to *cvx*.

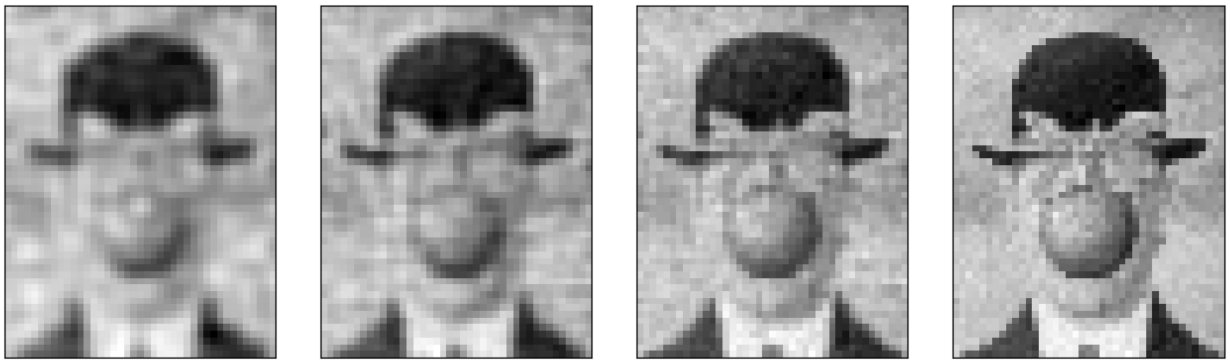

Figure 3: The thresholds percentages starting from the left and moving right are as follows: 5%, 10%, 20%, and 40%.

Turning to convex optimization techniques for image recovery, using r values of $0.2, 0.4$, and 0.6 resulted in Figure 4 with the reconstruction of the image preformed three times using each value of r. This is to ensure that there is a "fair" distribution of observed pixels to reconstruct the image from. The technique was then employed on the unknown image's B matrix and y vector which resulted in Figure 5 with similar appearance to $r = 0.6$ but actually has a value of $r = \frac{2000}{2500} = 0.8$.

5 Summary and Conclusions

Using the Discrete Cosine Transform for image compression has noticeable drawbacks. As more coefficients are utilized, the image becomes sharper and less symmetric, but this comes at the cost of image compression. The first image of Figure 3, representing 5% of DCT coefficients, has the leaves of the apple represented by strong circular and symmetric shapes. Viewing this image, a layperson might recognize the painting, but they wouldn't be able to reliably view the details of the image. Many features of the 5% reconstruction are open to interpretation: the leaves can be viewed as spectacles; the shadow of the apple as a disturbing smile. Increasing to 10% removes some of the blurriness and symmetry of the 5% reconstruction but retains many of its flaws. Retaining 20% and 40%

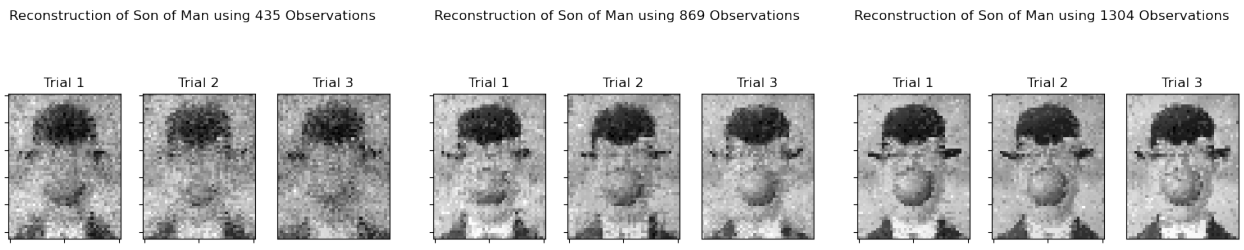

Figure 4: The number of measurements given r of each set of trials starting from the left and moving right are as follows: $r = 0.2$ with 435 observations; $r = 0.4$ with 869 observations; and $r = 0.6$ with 1304 observations.

of coefficients produces results that almost identically matches the original down-scaled image, save some minor gradient issues in the background. All of these reconstructions, however, struggle to capture the titular man's eye which is an important part of the image as a whole. DCT and other signal compression messages might excel at broad reconstruction, but finer details, such as a small eye peaking out from an apple, are overlooked.

Turning to image recovery using convex optimization, for $r = 0.2$, the majority of the face/apple is covered in static while the hat and suit are well defined. The accuracy in reconstructing his suit and bowler hat is understandable given that both articles of clothing are comprised of uniform shapes and colors that cover much of the canvas. This means that pixels from his hat or suit are more likely to be observed and then used to reconstruct the image. Likewise, since the face and apple are complex due to their shading and anatomy, it is expected that both will be unintelligible at a low r value. Trial 3 for this value of r resulted in a noticeably poor reconstruction while the previous 2 trials were much better with Trial 1 being the best. For $r = 0.4$, the apple itself is much more visible but the delicate leaves are still inseparable from the rest of the hidden face. For Trial 3 of $r = 0.4$, the apple and suit are defined very accurately implying that many of the randomly selected pixels came from those two regions. For $r = 0.6$, now all the main features of the painting are visible, but along the gradient of the apple's shadow, poor performance is still very visible. Also, many pixels of the background are inverted, e.g., if they are supposed to be lighter, they instead are darker, contributing to a speckled or salt-and-pepper look to all of the reconstructions.

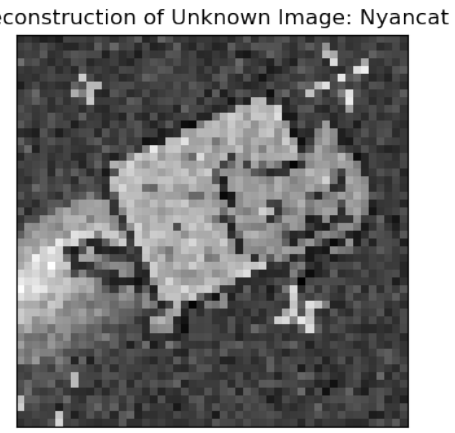

Figure 5: The reconstructed unknown image is a picture of Nyan Cat.

When this technique is applied to the unknown image set, a low quality Nyan Cat is reconstructed. Here, the background has the characteristic salt-and-pepper speckling while the right eye is smooshed and deformed. The gray-scale gradient for the rainbow coming out of its back, however, is of noticeably quality. Using convex optimization techniques, however, is extremely computationally intensive and is not viable as was implemented in this assignment. Other methods should be employed for image reconstruction instead.

6 Acknowledgements

The author would like to graciously thank his classmates for allowing him to discuss the problem with them as a means to help them and test his own understanding of the material.