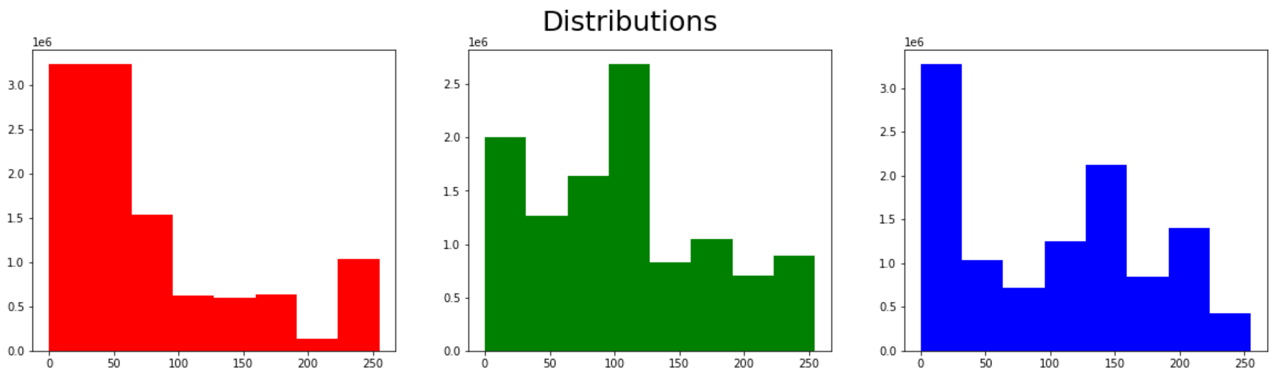
Emilio Padrón Molina

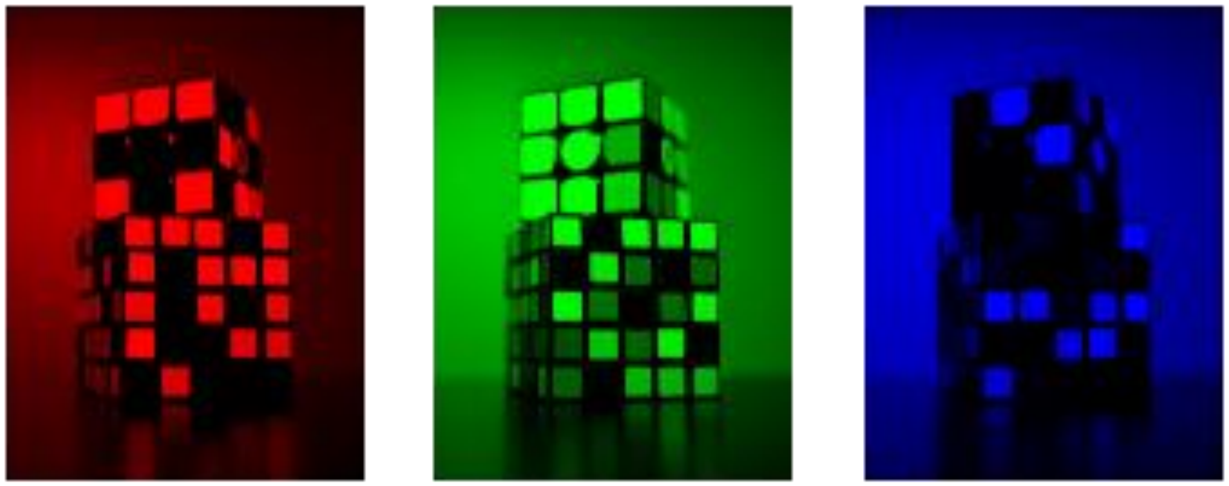
Image Compression using Non-Negative Matrix Factorization Report

Objectives and Description of the Dataset

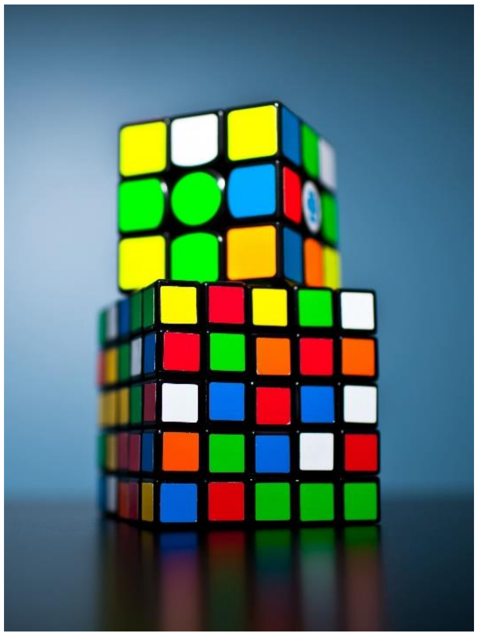
For this report, I decided to work with a colored image obtained from Google, and apply concepts of NMF dimension reduction to compress it and see its output per number of dimensions. The image selected for this project is of 3840x2880 pixels and, therefore, is of shape (3840, 2880, 3), where the last dimension represents the red, green, and blue channels (given that it is a colored image) where each value ranges from 0-255. These color values are distributed as follows:



which can also be observed through the image decomposition:



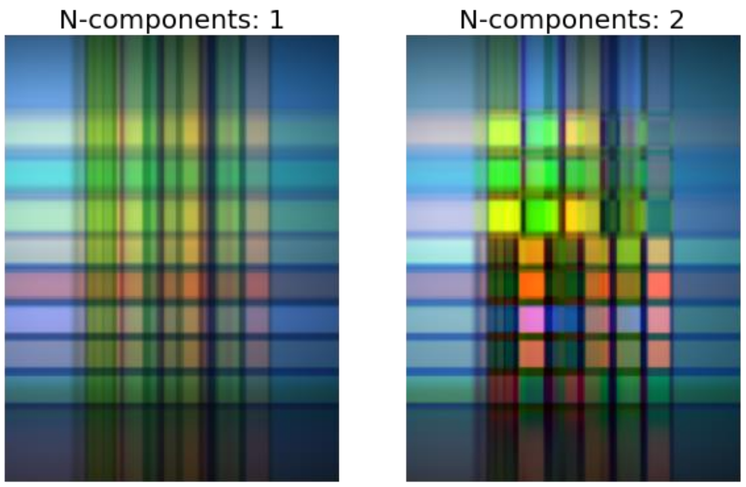
From the previous graphs, one can observe that the green channel is the one that appears to be more normally distributed while the red and blue channel appear to be rightly skewed (the red having a grater skew). By adding the three channels together, the original image can be rebuilt:

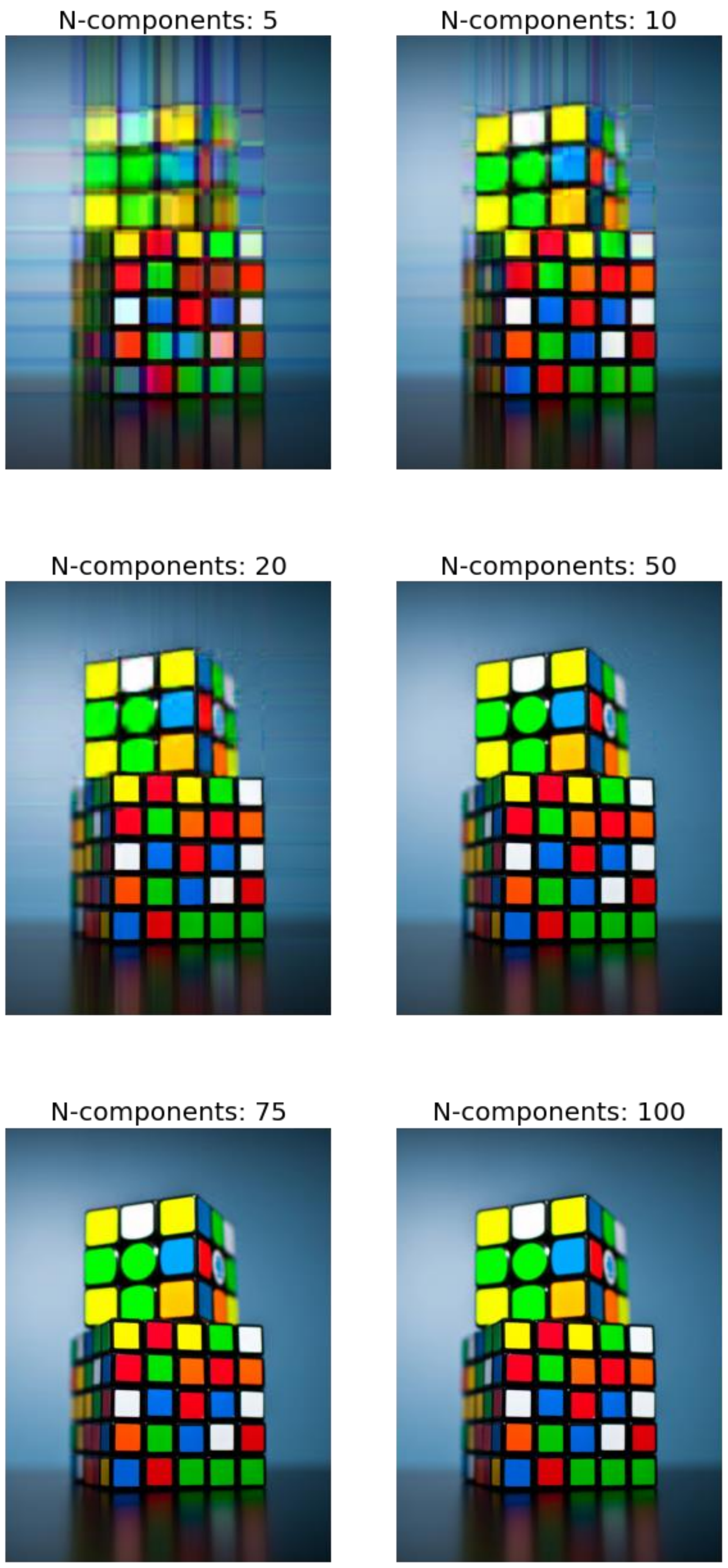


Data Modelling

To compress this image, it is necessary to scale each of the color components and then define a function, “reduced_image”, that asks for the number of components after the reduction and the scaled image. This function grabs each color channel and introduces it to the NMF (with 500 iterations) object from Scikit-learn and then calculates its *W* and *H* components. Next, these two matrices are multiplied for each case to generate the specific channel of the new image. As one may guess, the final new image is the superposition of the separate channels.

Furthermore, another function, “print_im”, can be programmed in a way that plots the compressed images outputted from the “reduced_image” function for different numbers of components. Using an array of 8 different n-components, one can see the results of differently compressed images:

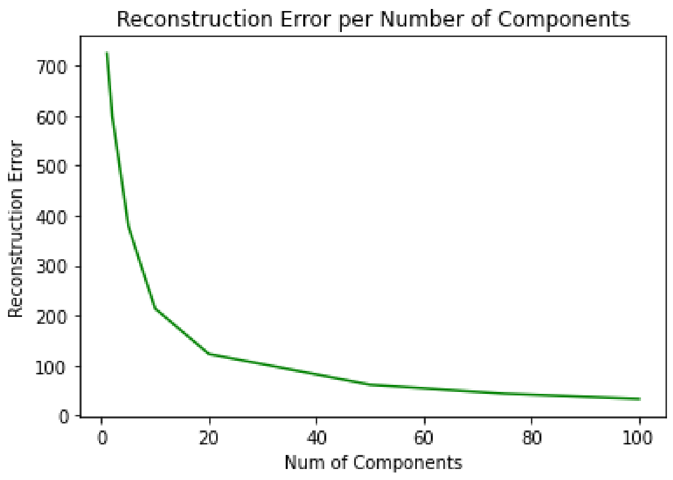



N-components: 5
N-components: 10
N-components: 20
N-components: 50
N-components: 75
N-components: 100

Errors

For each of the previous images, one can calculate the reconstruction error of each of the color channels and then average them out. This average reconstruction error per number of components can be put in a data frame and in a line-plot as follows:

Num of Components	Avg. Error
1	724.692749
2	598.786982
5	379.108525
10	214.237356
20	123.326383
50	61.553245
75	43.653959
100	33.267832



Insights and Analysis

In this case, one can see from the images and error that for more than 50 components, there isn’t a significant difference between the original and the compacted image. Therefore, the optimal compacted image using Non-Negative Matrix Factorization is the one that has around 50 components. This is because the error doesn’t significantly decrease after this point and the image itself proves to have a good resolution.

Finally, it is worth mentioning that to improve this result one can try applying the “print_im” function for other number of components around 50. Also, one can adjust other parameters such as the maximum number of iterations, the beta loss, or the initialization. Moreover, one can try other dimension reduction models, for instance a PCA model, to see if it provides better results than NMF.